Distributed Network Defense System

Yehuda Afek, Hacarmel 26, Hod-Hasharon, Israel, and
Anat Bremler-Barr, Hashomer 8, Holon, Israel

1. Abstract

A new concept and direction in the field of network and computers protection from cyber vandalism and Internet hackers is presented in this patent. In particular our mechanism is a flexible and robust method to protect servers and routers in the Internet from different distributed denial of service (DDoS) attacks, enabling continuous operation during an attack. Our method enables an ISP to provide protection against DDoS to all the servers residing in its autonomous system, thus relieving the individual servers from the need to provide protection from DDoS (over load and swamp) attacks. Our methods may protect any element in the network: computers, routers, servers and other elements. The invention has two major components, a guard machine and the concept of distributing the guard machines in key points in the Internet to achieve distributed defense against distributed denial of service attacks. In particular the idea is to place guarding machines all around a protected area of the network (e.g., an autonomous system), one guarding machine for each peering point of that area. The invention details the architecture, structure and operations of a guard machine, and the mechanisms and procedures that enable the distribution of the guards to protect particular victims in the network.

2. Introduction

Despite numerous distributed denial of service attacks that took place last year (with a surge of attacks on YAHOO, CNN, and many other major sites), there is still no known online solution that directly protects during an attack. Here we present a distributed defense system that does this, enabling the continuous operation of an attacked site while the attack is going on.

Our method of protecting victim elements in the network during an attack and enabling them continuous operation despite the attack follows these major steps:

a. **Detection:** We assume that an attacked victim has an automatic mechanism that detects and alerts when an attack begins. Such a mechanism is provided by different routers, firewall equipment and operating systems.

b. **Alert:** Upon suspecting that an attack has began an alert network activates a network of guard machines that are located at strategic points around an area of the network in which the victim resides. For example, the guards could be located around the autonomous system that hosts the victim.

c. **Divert:** In addition the alert network invokes a rerouting mechanism that ensures first that all the traffic destined to the victim is diverted to the guards and second that no packet (message) reaches the victim unless it passed through a guard.

d. **Sieve:** When an alert arrives at a guard and the victim traffic is being received, the guard sieves the traffic to sort out the "bad" packets and pass on to the victim only the "good" traffic. The architecture and operation of the special purpose guard machine is the second part of this patent and has the following four major components (see Figure 1):

1. **Anti-spoofing:** An anti-spoofing module that authenticates and verifies for each flow (<source-address, source-port, destination-port> triplet) that a real process at the host with that source-address and behind that port-number has initiated this flow.

2. **Statistics:** A module that detects and singles out flows (Source IP addresses or subnetworks) with outstanding behavior. The identity of these flows is passed to a filter.

3. **Filter:** A module that blocks any packet originating from an IP address or subnetwork that was identified in the previous step as a source of malicious traffic.

4. **Ingress filtering:** The guard machines interact with its neighboring routers to enable an effective usage of the ingress filtering feature. The routers do not always know which flows they may block because of route asymmetry. The guards analysis of the traffic both at normal operation and during an attack would to pin point which IP addresses and addresses blocks may be purged.

5. **Termination detection:** All the guards participate in a fourth module which is a distributed algorithm they run to cooperatively decide when an attack has stopped and the victim may return to peaceful operation mode. This last transition has to hand over the good connections from the guards to the victim.

Packets flow through a guard machine by first passing through the third component, then through the first and finally through the statistical module.

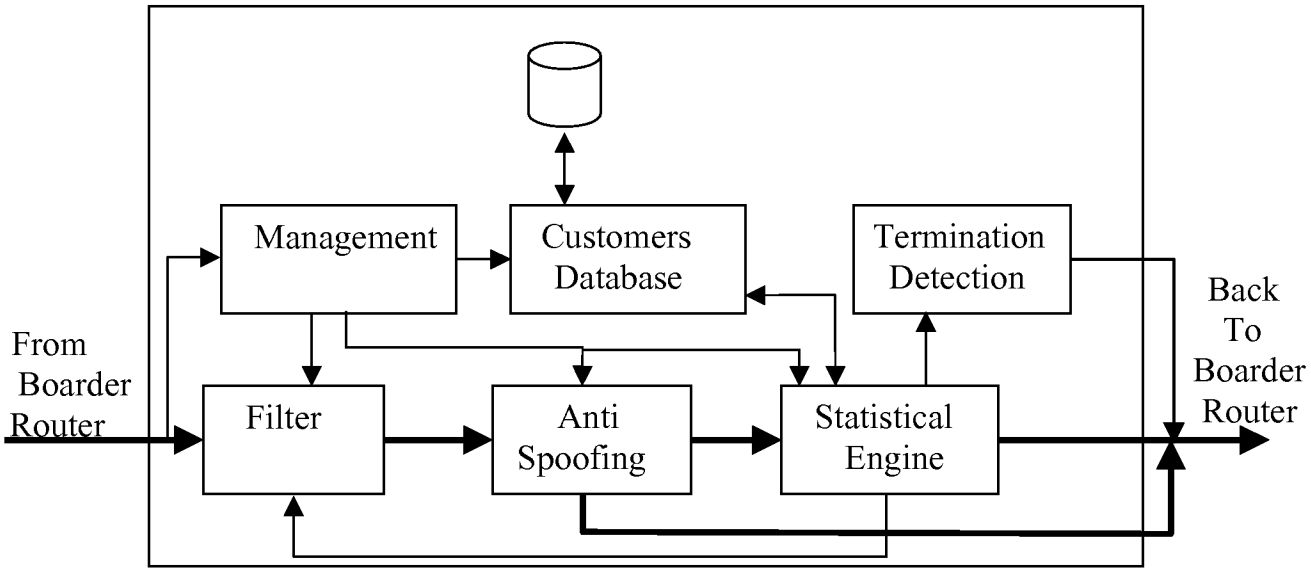


Figure 1: High level block diagram of a guard machine. Thick lines represent the flows of victim traffic through the machine while the thinner lines represent control and other information paths. Customers database maintains statistical information about each potential victim traffic patters (each customer is a potential victim that the guard would protect).

In addition we describe how to deal with the open connections while transiting from normal operation to protect mode, and how to deal with the open connections while transiting from protect mode back to normal mode (after an attack has been stopped).

The diversion method that passes all the victim traffic, during an attack, through the guard machines is described in the next section. In Section 4 the antispoofing module is described and the .....

3 Activating the Guards system

3.1 Redirecting traffic to NetGuards

Upon suspecting that an attack is being mounted on it, a victim alerts the guards through a communication channel supplied by the backbone provider. For example by sending authenticated messages to the NOC (Network Operations Center) from which the message is relayed to the guards (SNMP and other network management means may be used instead). The alert message contains the identity of the victim machines (which includes their IP addresses). At this point the victim enters the "protected" mode.

In this "protected" mode all the traffic flows to the victim have to be diverted such that:

1. All the traffic whose destination is the victim, from either outside the autonomous system (AS) that hosts the victim or from inside the AS, is redirected to the guards.
2. All the traffic that reaches the victim must have passed through one of the guards, i.e., no traffic can reach the victim without going through the guards.

In this section we describe one possible architecture and mechanism to achieve the above goals.

Two IP addresses are associated with each victim destination machine (server) the ***server (victim) public address*** and the ***server (victim) private address***. The *server public address* is the IP address of the server which is published all over the Internet through the DNS system. The *server private address* is an IP address used solely to transfer packets between the guards and the victim, while the guards protect the victim. Therefore, the *server private address* is recognized only by either router interfaces that are connected to internal links of the hosting AS backbone, or to the guards, or to the victim interface. All other interfaces, such as, connections to links that come from outside the AS, or links that are connected to other customers of the AS (stub networks), discard any packet whose destination is the *server private IP address* (easily and efficiently achieved by using the CEF mechanism of the routers). (See figure 1). This ensures that no hacked daemon can generate traffic to the *victim private IP address.*

To achieve the above setting all the interfaces that are connected to external links, i.e., links that connect to either other networks (AS's) or to external hosts and customer networks, are permanently programmed to discard traffic destined to the *server private IP address.* In normal operation when no attack is being mounted, the victim declares itself to be at distance zero from both the *server private IP address* and the *server public IP address*. This causes the routing protocol to set entries in the forwarding tables in all the AS routers, to forward messages destined to either address to the victim (which is now not a victim) machines.

To divert *public IP address* victim traffic that arrives from outside the hosting AS during an attack we notice that all such traffic must pass through one of the border routers, i.e., a peering or NAP, BGP routers. A guard machine is placed in each entry next to the boarder routers at this point. Upon receiving the alert of a possible attack on a victim all these boarder routers are set to forward all the traffic arriving from out of the network (AS) and whose destination address is the *victim public IP address,* to the guard machine which is placed next to them. This is easily achieved by injecting an update to the boarder routers, updating their policy routing mechanism. Updating the policy routing mechanism, gives us the ability to change the routing behavior without degrading the border routing performance[1]. In effect the guarding machines become a TCP proxy for the victim. All the traffic returning from the victim to trusted clients is passed through the corresponding guarding machine.

[1] Unlike access list, that required filtering every packet, and hence degrading the router performance, Policy routing doesnot harm the routing performance.
To understand this, we briefly explain the look up process in today routers. Most of the routers use Cisco Express Forwarding, or some equivalence mechanism. Using FEC, every interface has a cache where it store the information about the next hop for the last packets that arrive throw this interface. When packets arrive to the interface card and the destination is not store in the cache, a new forwarding process is done. This process is done in the central unit that does the lookup process for all the interfaces. This process take into account the routing policy that can be defined per interface. This operation is done rarely and in almost all of cases, the lookup operation use the cache information. Hence the impact of the degrading in the forwarding time is minor.

To divert victim *public IP address* traffic that originates inside the hosting AS to the internal or boarder guards, one could use a similar mechanism. That is, to inject when the alert is received, the desired routing information into all the routers. However, this requires updating the policy routing of all the routers in the AS. In many cases (large networks), it is more beneficial to use a different method, based on a simple routing manipulation. In this method, when the victim suspects that an attack is being applied, it declares itself to be at a large distance from the *server (victim) public IP address,* while the guards would start to declare that they are at distance zero (or close to zero) from the *server (victim) public IP address.*[2] This routing updates quickly spread in the AS network, using the standard routing protocol (usually a link state type of protocol), e.g., OSPF, EIGRP or RIP[3]. Thus within seconds from these declarations all the victim traffic is automatically diverted to the guards.

When the guards decide (see how below) that the attack has terminated they send an appropriate message to the victim machine. At the same time they reverse the above settings, that is, they stop declaring that their distance from the *server (victim) public IP address* is zero, while the victim starts declaring again that it is at distance zero from its public IP address.

Notice that in the "protect" mode several guards may all claim to be at distance zero from the *victim public IP address.* This divides the AS into clusters, such that packets with this destination address in each cluster are routed to the guard residing within that cluster. However, there might be routers on the boarder between two or more such clusters with equal distance to two or more guards. This may introduce routing instability, where some packets of a flow go to one guard and some packets go to the other guard. First notice that this effects only victim traffic that originates inside the hosting AS. The victim traffic that arrives from outside the hosting AS is treated by the guard at the entry point which acts as a proxy for that traffic. Thus outside traffic would suffer from route flapping only if these flapping are introduced by BGP, which is very rare. To avoid the flapping of victim traffic originating inside the AS, we set each guard to declare that it is at a very small but different distance from the victim *public IP address.* This small perturbations ensure that no router would be at equal distance from two guards. The exact calculation of this perturbation is automatically calculated given the ISP map of its backbone.

[2] In some case in order to handle the volume of the intra network traffic, it may beneficial to use not one NetGuards, but a farm of NetGuard However, one should noticed that the problem of attacks, and special spoofing attack, in most of the cases is harder when the attack is originated from outside the network. When the attack is originated from inside network, there is full information and management of the network. Hence ingress and egress filtering can be used, for dealing with spoofing attack. In cases when the origins of the attack are known, one can more easily stop the attack, by disable the origins of the attack.

[3] Unlike BGP, these routing protocols adapt very quickly to topological changes, thus correcting the forwarding tables in all the routers in hundreds of milliseconds.

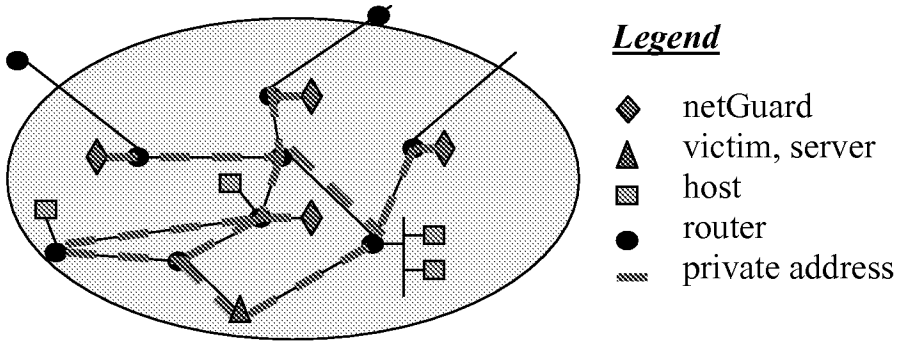


Figure 2: The *victim private address* is known only to trustable parts in the networks, i.e., the interfaces of routers that connected to another router or to netGuards. The routes in the network where the *victim private address* is known is marked by dashed red lines.

4 TCP Anti-spoofing

We describe here an anti-spoofing techniques which is TCP oriented. The basic principle of the idea was presented before by Checkpoint and Cisco (TCP intersept). The innovation here is in the way it is combined with the other mechanisms and its employment on the boarders of the hosting AS. This anti-spoofing mechanism authenticates the genuine of the source address of the flow, based on the SYN mechanism of TCP. Doing so each guard in effect becomes a very low level TCP proxy for the victim. When a client wishes to open a TCP connection with a server, it sends a SYN request, notifying the server about its attempt to open a new connection. The server authenticates the client source address by sending the client a random number. Then, the server waits to receive this random number back from the source. A spoofed source cannot repeat the number, and hence any connection between the server and a spoofed source is dismissed. (see figure 2).

The SYN mechanism or the connection establishment (three way handshake) is also one of the known denial of service attack methods. In this attack a huge number of spoofed SYN-requests are being sent to the server. Each such request must be buffered and kept by the server for a period of time (30 seconds by the standard) until its corresponding SYN-ACK is received. The SYN-request buffer at the server overfills which at worse brings the server down and at best causing the server to loose good genuine requests to open new connections.

Each of our guard machines is a special purpose machine that among other things performs the connection establishment on behalf of the victim. Being special purpose it can handle a huge number of connections at very high speeds (supporting OC-192 lines). Moreover, the distributed architecture of our system distributes the load among the different guard machines.

In the next two figures we show the sequence of messages during the three way handshake. The first figure, Figure 3, shows the normal sequence of messages during the three way handshake between a client whose IP address is 12.12.12.12. and a server whose IP address is 10.10.10.10. In Figure 4 the same process is performed but now the SYN request message is intercept by the guard machine which then performs the three way handshake on behalf of the server. Only after the guard machine receives the correct SYN-ACK message from the client it opens the corresponding connection with the server and starts to function as a proxy between the client and the server.

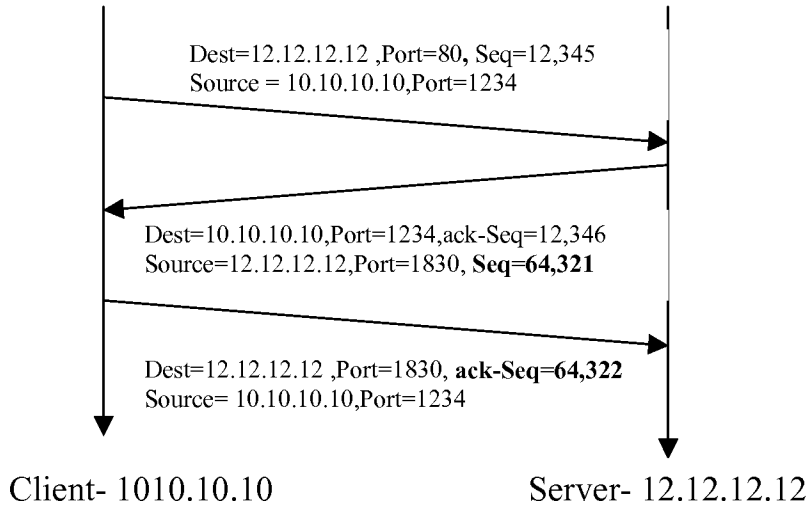
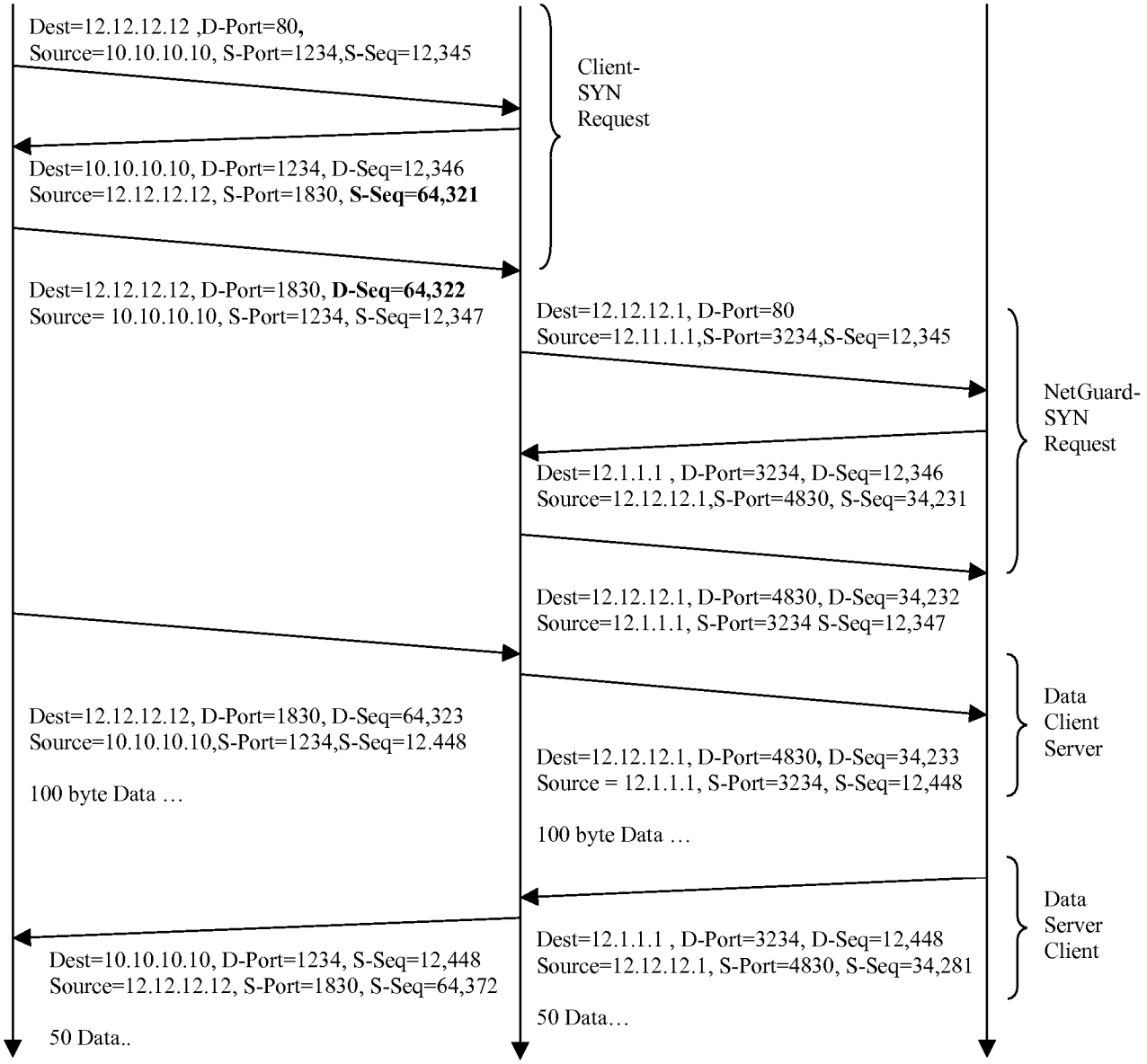


Fig 3: The SYN request.



Client- 10.10.10.10

NetGuard – 12.1.1.1
Play the rule of 12.12.12.12
12.12.12.12 is the server public address

Server- 12.12.12.1
this address is the private address of the victim

5. Ingress filtering

All bla bal bla bla bla authentication and was not stopped by the

6. Attack Identification, Recognition and Isolation via the Statistical Recognition Unit.

All the victim traffic that has passed the anti-spoofing authentication and was not stopped by the filter flows through the statistical unit. The statistical unit analyzes the traffic and identifies malicious sources (i.e., compromised sources), and provides operational rules for blocking the attack without disturbing innocent genuine traffic. The basic principle behind the unit's operation is that the pattern of traffic originating from a black-hat daemon drastically differs from the pattern generated by such a source during normal operation. In contrast, traffic patterns of "innocent" sources during an attack resemble their traffic at normal times. This principle is used to identify the attack sources and provide guidelines for their blockage by either the filter or the access lists of the routers. For example, the volume of a traffic from an attacking daemon, the distribution of packet sizes, port numbers, the distribution of the packets inter arrival times, and the ratio of inbound and outbound traffic are all parameters that may indicate that a source (client) is an attacking daemon.

The statistical unit has two major tasks:

1. Learning the traffic patterns during normal operation, i.e., when no attack is being mounted. These patterns are used while defending a victim during an attack to compare with the actual traffic in order to distinguish the malicious traffic from genuine traffic. We consider three possible ways in which this learning can be done: (1) Using the routers NetFlow data, (2) Analyzing the server logs at the victim server, and (3) Analyzing the potential victim traffic at the guard by having the traffic diverted to the guard from time to time for randomly sampling it.
2. Monitoring the victim traffic during an attack to identify and isolate the malicious traffic from the good genuine traffic. The identity of the attacking host is then given to the filter or the neighboring routers that would then drop any packet arriving from that host.

5.1 Network flows and traffic classification

The basic element studied by the statistical unit is a flow. Each flow is a sequence of packets belonging to the same connection. In the most general way a flow is identified by the following parameters: Source IP address, Source port, Destination port, Protocol type, time of day and day of week of connection creation. The destination IP address is implied since we collect all the information per destination address. For each such flow the traffic volume is registered.

Keeping all of the above information is infeasible since it requires an unacceptable amount of memory. However we employ learning methods to study the basic characteristics of the traffic destined to each destination and keep these key

parameters succinctly, in an efficient way. Essentially the learning method studies the typical behavior of groups of users that interact with the destination. For example, a typical web site is accessed either by individual users sitting behind a host (pc), by a group of users sharing one multi user time sharing host, or by a group of users sitting behind a proxy. For each such group its typical behavior is studied. Other types of users are possible such as web crawlers (for search engines) and monitoring servers such as keynote (www.keynote.com). Furthermore, for the largest group, i.e. the group of individual users, their identities (IP address) is not kept. Each source which is not included in the other groups is assumed to be an individual source. On the other hand, for the group of proxy machines that access the destination, the individual IP address of each is kept in a trie like data-structure. For other groups of users only their IP address may be needed since their traffic would be blocked from the beginning during an attack. Henceforth, the rest of this section considers types of users and the characteristic of flows originating from such users.

The basic parameters characterizing each user group are:

1. **Traffic volume distribution**: These include the **mean** and **variance** of the traffic such a user generates.
2. **Port numbers distribution**: Source port number distribution, and destination port number distribution.
3. **Periodicity:** Sources will be examined for the periodicity of their requests. It is likely that malicious sources act in a relatively periodic manner, while innocent sources act not in a regular manner.
4. **Packet Properties:** The distribution of packet sizes.

5.2 Learning Traffic Characteristics

There are three possible ways, that we consider, to learn and analyze the traffic characteristics of a particular target:

1. Sampling a **fraction** α of the **packets** $(0<\alpha\leq 1)$ traversing the lines on route to the target and then classifying the packets according to the flow id and time of day and day of week.

   Notice that setting α=1 requires the unit to process every packet and thus imposes high load on it while providing the best statistical measure. Lower values of α reduce the load posed on the unit while potentially somewhat degrading the statistical measure. The fraction α, therefore, will be a parameter that will be set so that enough statistical knowledge can be gained without over-loading the system.

2. Utilizing **server logs** collected by the defended target. These typically contain information about the activity being applied on the target. For example, WEB sites, which are likely to form the main body of potential targets, keep logs that record all the document requests sent to the site (including their source address, time of the day and other parameters). Processing of these logs by ZZZ yields a very accurate measure of the statistics of network flow volumes (measured in packets per second, as in a) above). The potential draw backs of this method are first that being collected at the target it is not immediately clear which information is relevant to which guarding point, and second, the pattern seen by

the target may be slightly different from the pattern seen at the network boarder. However neither is a real problem and the first one may be a feature, since network routes change and the traffic may enter the network from a different point in any event.

3. Analyzing netflow data collected from the appropriated routers. This option requires the backbone provider to enable netflow and process it with our learning applications. This method has some limitation but none seems prohibitory. The limitations are that netflow aggregates information for each flow in intervals of a few minutes (typically 5 minutes intervals), and in this intervals it does not maintain the sizes of individual packets. Rather, it counts the total number of packets and bytes passed in this interval for each flow.

5.3 Traffic Monitoring and Analysis at Attack Time

In attack time while the guard machine defends a victim it monitors the victim traffic, classifies its traffic (incoming and outgoing) and compares the traffic to the normal traffic in order to detect the malicious traffic. Notice that during an attack information is collected only on the current flows. The information about well behaving flows is not kept more than small number of minutes.

1. **Online traffic volume collection at attack time:** This module collects the statistics of the traffic destined to the target(s) in attack time. Notice that in this sense, its measures are similar to the measures collected in approach 1i above. The classification of the traffic, in general, is similar to that conducted in the learning phase but may be controlled/guided by external intervention. Such intervention is enacted if some additional knowledge on the attack type is gained from other sources (e.g., human-aided identification) and can be utilized by the unit.
2. **Attack Analysis:** Is conducted in attack time and is responsible to compare the statistical data learned with the current traffic volume and generate rules for traffic blockage. The output of this unit, in general, will consist of a list of items for each of which three parameters will be provided:
   a. **Network flow**, identified by a combination of source IP address (can be prefixed), destination IP address, destination port number, protocol type (one may consider blockage that disregards port numbers, i.e., all the traffic originating from a compromised IP address, be it a proxy or a host).
   b. **Duration,** identifying the duration for which that class will be blocked.

   The analysis is based on the statistical parameters of the data and aims at keeping the target traffic at normal loads by blocking the most "suspicious" traffic streams. Blocking rules are based on maximizing the likelihood of blocking malicious traffic while minimizing the likelihood of blocking innocent traffic.

5.3.1 Statistical Recognition of Data "Innocence"

NETGUARDS uses two major properties of network flows to identify malicious traffic: a) Traffic pattern, and b) Traffic volume. Below we describe the recognition approaches based on these factors.

5.3.1.1 Recognition of Traffic Pattern

Several aspects of traffic pattern are examined:

1) **Source "IP geography" proximity**: Sources will be classified into classes that resemble the "IP geography", that is IP addresses that reside on neighboring networks (using IP address prefix) are classified in the same class. A class that generates a relatively large volume of requests is suspected as being malicious. Notice, that such "malicious classes" are likely to form if the attacker planted a collection of daemons in the same network, and this network does not use a proxy.

2) **Periodicity:** Sources will be examined for the periodicity of their requests. It is likely that malicious sources act in a relatively periodic manner, while innocent sources act in more irregular pattern.

3) **Packet Properties:** Sources will be examined for repetitive properties of their packets. For example the distribution of packet size. It is likely that malicious sources generate packets of identical properties (e.g. – all packets of same size) while innocent sources will generate packets of more random nature. Other properties include port number distributions.

5.3.1.2 Recognition of Traffic Volume

Traffic volume recognition is used to identify malicious sources that transmit **large volumes** of data which **significantly differ** from their normal volume. Specifically, we classify Internet data sources to *small sources* and *large sources.* The former relates to individual IP addresses whose traffic volume is normally tiny. The latter relates to Proxy traffic or Spider (Crawler) traffic[4] whose volume is drastically higher.

ZZZ keeps individual volume parameters for each of the large sources. Individual parameters are not kept for the small sources; rather a single fixed small number (related to their mean volume averaged over all these sources) will be recorded. At attack time the traffic volumes of individual flows will be measured and compared to their recorded volume. Flows whose volume drastically differs (upwards) from their recorded measure are marked as being malicious.

The mathematical formulation of this procedure is as follows: Given are K classes of flows, indexed *1, 2, ..., K,* and characterized by the mean (μ_i) and the

[4] The traffic volume resulting from a Spider access is normally higher than that of a single human user, especially on large WEB sites. The reason is that a spider scans the whole site, leading to hundred or thousands of requests while a human client requests tens of pages or less, on average.

variance (σ_i) of their learned volume, and by their current volume (X_i). We would like to identify the classes with the largest deviation from their corresponding expected volume. Let $Y_i = (X_i - \mu_i)/\sigma_i$. We will sort the classes by the value of Y_i and recommend blocking the classes with the largest values of Y_i.

5.3.2 Time accumulating traffic volume recognition and "controlled" denial of service

It is important to recognize that the effectiveness of volume recognition increases with the time duration along which it is implemented. This is correct since the variance of total data volume generated by a source during a period of duration *T* decreases in *T*. For example, it is expected that the average amount of traffic generated by a small source during a period of 1 hour will be *very small.* However, at certain epochs, it is expected that the average amount of traffic generated by the same source during a period of 1 minute can be rather large. (up to 60 times larger than that of the 1 hour average).

For this reason ZZZ will implement the following unique recognition and traffic screening mechanism. For source *i*, let $S_i(t)$ denote the amount traffic generated by the source during the interval *(0,t)* (where we assume that the attack starts at time 0). We then set at time *t:* $X_i(t) = S_i(t)/t$ and apply the above screening mechanism.

This mechanism has the following properties:

1. For a small value of *t* (that is, at the first few minutes of the attack) a sophisticated attacker might cause significant number of innocent users denial of service. This is due to the fact that the attacker may inflict a load that resembles that of an innocent client, and thus the attacker is not distinguishable from the innocent client. At this stage, ZZZ may block some innocent clients and some attackers. Using this action, for a short period of time, some innocent clients may be denied of service but ZZZ protects the site from going down.
2. As *t* increases more and more malicious sources are identified and blocked and fewer innocent sources are blocked. This is since the malicious sources have posed large amount of accumulated load. Thus as time progresses less and less innocent clients are denied service. In fact, after relatively short period all malicious sources will be denied service while the innocent sources will receive full regular service.

Example: Consider the traffic volume generated on the web site of the Nagano server (Feb 98). It had 11,665,713 requests made over a period of 24 hours by 59,582 clients. Assuming uniform distribution of clients over the day, this implies about 2500 clients per hour and 500 clients per 12-minute interval. An attacker who uses 500 sophisticated daemons (which imitate a normal client) will look innocent at the first 12 minutes interval. At this period ZZZ will block 50% of the innocent clients and 50% of the daemons. However, after 24 minutes the daemons will generate significantly more traffic than an innocent client and thus almost all of the traffic blocked will be that of malicious daemons.